

Neo4j Plugin

Description

The Neo4j plugin enables lightweight access to database functionality using [Neo4j](#). This plugin does **NOT** provide domain classes nor dynamic finders like GORM does.

Installation

The current version of `griffon-neo4j` is **0.4**

To install just issue the following command

```
griffon install-plugin neo4j
```

Usage

Upon installation the plugin will generate the following artifacts at `$appdir/griffon-app/conf`:

- `Neo4jConfig.groovy` - contains the database definition properties.
- `BootstrapNeo4j.groovy` - defines init/destroy hooks for data to be manipulated during app startup/shutdown.

A new dynamic method named `withNeo4j` will be injected into all controllers, giving you access to a `org.neo4j.graphdb.GraphDatabaseService` object, with which you'll be able to make calls to the database. Remember to make all calls to the database off the EDT otherwise your application may appear unresponsive when doing long computations inside the EDT.

This method is aware of multiple datasources. If no `datasourceName` is specified when calling it then the `defaultDataSource` will be selected. Here are two example usages, the first queries against the default datasource while the second queries a datasource whose name has been configured as 'internal'

```
package sample
class SampleController {
    def queryAllDataSources = {
        withNeo4j { dsName, graphdb -> ... }
        withNeo4j('internal') { dsName,
graphdb -> ... }
    }
}
```

This method is also accessible to any component through the singleton `griffon.plugins.neo4j.Neo4jConnector`. You can inject these methods to non-artifacts via metaclasses. Simply grab hold of a particular metaclass and

call `Neo4jConnector.enhance(metaClassInstance)`.

Scripts

- **create-relationship-type** - creates a new enum that implements `RelationshipType` at `src/main`.

Configuration

Dynamic method injection

The `withNeo4j()` dynamic method will be added to controllers by default. You can change this setting by adding a configuration flag in `Config.groovy`

```
griffon.neo4j.injectInto = ["controller",  
"service"]
```

Events

The following events will be triggered by this addon

- **Neo4jConnectStart[config, databaseName]** - triggered before connecting to the datastore
- **Neo4jConnectEnd[databaseName, db]** - triggered after connecting to the datastore
- **Neo4jDisconnectStart[config, databaseName, db]** - triggered before disconnecting from the datastore
- **Neo4jDisconnectEnd[config, databaseName]** - triggered after disconnecting from the datastore

Multiple DataSources

The config file `Neo4jConfig.groovy` defines a default `dataSource` block. As the name implies this is the `dataSource` used by default, however you can configure named `dataSources` by adding a new config block. For example connecting to a `dataSource` whose name is 'internal' can be done in this way

```
databases {  
    internal {  
        params = [:]  
        storeDir = 'neo4j/internal'  
    }  
}
```

This block can be used inside the `environments()` block in the same way as the default database block is used.

Example

A trivial sample application can be found at https://github.com/aalmiray/griffon_sample_apps/tree/master/persistence/neo4j

History

Version	Date	Notes
0.4	10-21-11	Release sync with Griffon 0.9.4
0.3.1	11-08-10	Fix a metaclass problem when injecting dynamic methods
0.3	10-27-10	Release sync with Griffon 0.9.1
0.2	07-22-10	Release sync with Griffon 0.9
0.1	03-25-10	Initial release